Wireless Machine Sentinel

Aikasynkronoinnin kehitystyö

Jarno Ruutinen

Opinnäytetyö

Ammattikorkeakoulututkinto

Savonia
ammattikorkeakoulu

SAVONIA-AMMATTIKORKEAKOULU **OPINNÄYTETYÖ**
Tiivistelmä

Koulutusala Tekniikan ja liikenteen ala	
Koulutusohjelma Elektroniikan koulutusohjelma	
Työn tekijä(t) Jarno Ruutinen	
Työn nimi Wireless Machine Sentinel - Aikasynkronoinnin kehitystyö	
Päiväys 13.5.2011	Sivumäärä/Liitteet 41 + 0
Ohjaaja(t) yliopettaja Väinö Maksimainen, yliopettaja Arto Toppinen	
Toimeksiantaja/Yhteistyökumppani(t) Honeywell Oy	

Tiivistelmä

Projektin tavoitteena oli suorittaa Wireless Machine Sentinel -verkon kattava aikasynkronointi, jonka tarkkuus olisi 1 ms. Aikasynkronointi tarvitaan, koska järjestelmän toiminta perustuu yhtä aikaa mitattujen tietojen keskinäiseen analysointiin. Synkronointi pyrittiin toteuttamaan ilman muutoksia verkossa toimiviin WLAN 802.11 -tukiasemiin, joten projektissa päädyttiin jo aiemmin käyttämään apuna tukiasemien tasaisin väliajoin toimittamaa Beacon-kehystä.

Työn toteutusosassa keskityttiin aikaleimaamisen toteuttamiseen mittausasemalla sekä projektissa jo aiemmin toteutetun aikasynkronoinnin parantamiseen sekä synkronoinnin testaamiseen. Projektissa onnistuttiin pääsemään vaadittuun tarkkuuteen yhden tukiaseman kokoonpanolla.

Avainsanat
WiMS, G2C501, aika, synkronointi, WLAN, Beacon

Julkinen

SAVONIA UNIVERSITY OF APPLIED SCIENCES **THESIS**
Abstract

Field of Study Technology, Communication and Transport			
Degree Programme Degree Programme in Electronic Engineering			
Author(s) Jarno Ruutinen			
Title of Thesis Wireless Machine Sentinel - Development of Time Syncronization			
Date	13 May 2011	Pages/Appendices	41 + 0
Supervisor(s) Mr Väinö Maksimainen, Principal Lecturer and Mr Arto Toppinen, Principal Lecturer			
Project/Partners Honeywell Oy			

Abstract

The aim of this thesis was to accomplish network wide time synchronization in the Wireless Machine Sentinel network. The solution had an accuracy requirement of 1 ms. Time synchronization is needed because the system needs to make time dependent analysis from measured data. The time synchronization method was aimed to be modification-free for WLAN802.11 access points, so time synchronization was based on the Beacon-frame, which is sent at constant intervals by all access points.

The implementation focuses on timestamping at the measurement station, how the previous synchronization scheme could be improved, and also how to test the accuracy of synchronization. The accuracy requirement of the solution was met successfully using one access point.

Keywords
WiMS, G2C501, time, synchronization, WLAN, Beacon

Public

ALKUSANAT

Ensimmäiseksi haluaisin kiittää yliopettaja Arto Toppista työn ohjaamisesta, G2 Microsystems-asiantuntija-avun hankkimisesta sekä haastavan aiheen tarjoamisesta. Haluan myös kiittää yliopettaja Väinö Maksimaista työn ohjaamisesta. Kiitokset myös kaikille Honeywell-tuotekehityksen henkilöille, joilta sain apua ongelmatilanteissa ja tiedon puutteessa. Lopuksi haluaan kiittää perhettäni opiskeluaikanani saamastani tuesta.

Kuopio, 13. toukokuuta 2011

Jarno Ruutinen

SISÄLTÖ

LYHENTEET, SYMBOLIT JA KÄSITTEET

MS	Machine Sentinel
WiMS	Wireless Machine Sentinel
QCS	Quality Control Solution
eCos	Embedded Configurable Operating System
µC/OS-II	Micro Controller Operating System – II
EPC	Electronic Product Code
GPIO	General Purpose Input / Output
HAL	Hardware Abstraction Layer
IP	Internet Protocol
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
SSID	Service Set Identifier
LLSQ	Linear Least Square
SLR	Simple Linear Regression
SDK	Software Development Kit
API	Application Programmers Interface
TSC	Timestamp Counter
TSF	Time Synchronization Function
RTC	Real Time Clock
STA	Synchronous Time Averaging
TBTT	Target Beacon Transmission Time
BSS	Basic Service Set
IBSS	Independent Basic Service Set
SoC	System on a Chip
ISM	Industrial, Scientific and Medical, maailmanlaajuinen radio-taajuuskaista, jonka käyttö ei vaadi erillistä lupaa.
DSSS	Direct-Sequence Spread Spectrum
ODFM	Orthogonal frequency-division multiplexing
FHSS	Frequency-hopping spread spectrum
LTS	Lightweight Time Synchronization Protocol
TPSN	Time Synchronization Protocol for Sensor Networks
NTP	Network Time Protocol
PTP	Precision Time Protocol
FTSP	Flooding Time Synchronization Protocol

1 JOHDANTO

Työ tehtiin Honeywell Oy:n Kuopion tuotekehitysyksikölle. Kehitystyön kohteena on Wireless Machine Sentinel -järjestelmän WLAN 802.11b-tekniikkaan perustuva langaton versio.

Projektin tavoitteena on toteuttaa koko Wireless Machine Sentinel -verkon kattava aikaleimaus 1 ms:n tarkkuudella, jotta mitattuja suureita voidaan käyttää luotettavasti ajan suhteen synkronisesti tehtävien analyysien lähteenä.

Wireless Machine Sentinelistä on olemassa versio, joka perustuu Honeywellin omaan OneWireless-tekniikkaan, mutta tässä työssä on keskitytty ainoastaan WLAN-tekniikalla toteutettavaan versioon. Aiemmin projektissa oli jo päädytty käyttämään kellojen synkronoinnin apuna Beacon-kehyksessä välitettävää TSF-aikaleimaa.

Tässä työssä keskitytään pakettien aikaleimaamisen toteuttamiseen leafNodella sekä perehdytään siihen, kuinka Beacon-kehyksen mukana toimitettua aikaleimaa voidaan käyttää tarkempaan aikaleimaamiseen.

2 MACHINE SENTINEL

2.1 Machine Sentinel

Machine Sentinel on yksi Honeywellin paperiteollisuudelle tarjoamista laadunvalvontajärjestelmistä. Machine Sentinel on koko paperikoneen kattava järjestelmä, jonka avulla voidaan tunnistaa ja paikantaa paperin laadunvaihteluun vaikuttavat tekijät. Machine Sentinel on saatavilla omana ratkaisunaan tai integroituna Da Vinci -laadunvalvontajärjestelmän kanssa. [1]

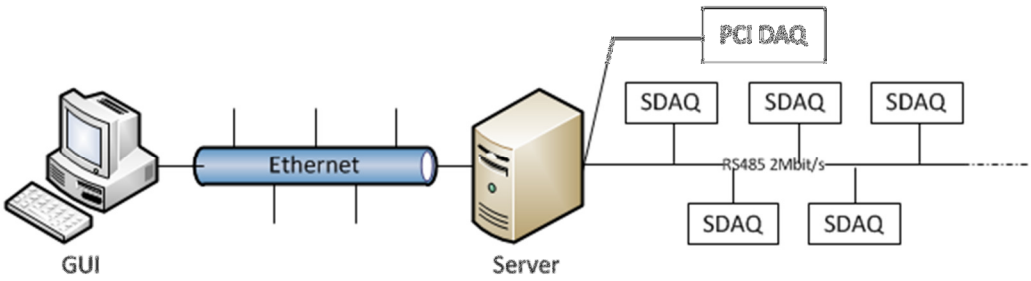
Machine Sentinel tarjoaa tärkeää tietoa kaikille paperikoneen kanssa työskenteleville, kuten koneenkäyttäjille, kehittäjille, kunnossapidolle, tehtaanjohdolle [1].

Järjestelmän tärkeimmät edut paperitehtaalle: [1]

- paperin laadun tasoittuminen ja paraneminen
- paperikoneen käyttökustannusten aleneminen
- paperikoneen käytettävyysajan kasvaminen
- koneenosien iän pidentyminen
- ratakatkojen väheneminen
- komponenttitoimittajien vertailun mahdollisuus
- varaosavaraston pienentyminen.

Machine Sentinelin toiminta perustuu paperin laatusignaalien mittaamiseen ja niiden vertaamiseen pumppuihin, siivilöihin, rulliin ja peitteisiin asennettujen takometrien pulsseihin. Mittaustulosten avulla voidaan tehdä ajan suhteen keskiarvoistamista ja näin paikantaa paperin laatuun vaikuttavat tekijät. [1]

Mittaustietojen perusteella voidaan määrittää taloudellisimmat käyttönopeudet ja parantaa näin monien komponenttien elinikää samalla säästäen varaosa- ja huoltokustannuksissa [1].



Kuva 1. Machine Sentinel -järjestelmä

Machine Sentinel koostuu palvelimesta, joka toimii mittausten vastaanottajana ja tietokantojen ylläpitäjänä. Operaattorin on mahdollista katsella mittaustuloksia ottaen yhteyttä palvelimeen kuvan 1 mukaisesti käyttämällä operaattorin koneella suoritettavaa GUI-sovellusta (Graphical User Interface).

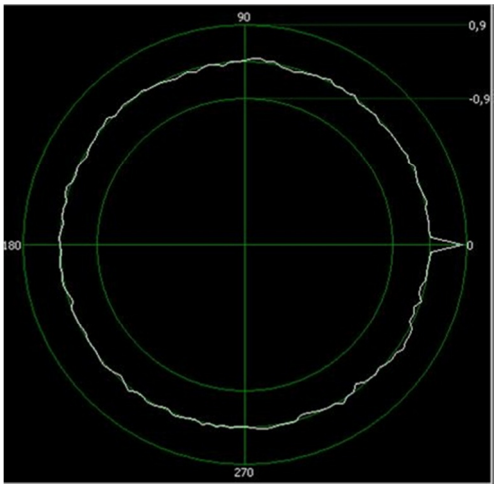
2.2 Sovellukset

Machine Sentinelin keräämien tietojen käsittelyyn on kehitetty sovelluksia, joista jokaisella on oma erityinen tehtävänsä: [1]

- Sheet Manager
  - mittaa lakanan laatuvaihteluja ja määrittää niiden vakavuuden ja lähteen.
- Press Manager
  - tarkkailee koneen prässien ja muiden niputettujen komponenttien tärinöitä tunnistaakseen laatuvaihtelujen lähteitä.
- Pulse Manager
  - tarkkailee paineen vaihteluja paperikoneen syöttöpuolella ja auttaa operaattoria tunnistamaan syötön vaihteluiden aiheuttajat ja näin takaamaan mahdollisimman tasaisen syötön.
- Bearing Manager
  - tarkkailee laakereita, vaihdelaatikkoja ja muita mekaanisen värähtelyn aiheuttajia tarjoten tarvittavat tiedot kunnossapidolle laakerien kunnosta, jolloin ylläpito ja ennakkohuolto helpottuvat.
- Trend Manager
  - mahdollistaa pitkän aikavälin tarkkailun mistä tahansa prosessin parametrista.

2.3 STA-analyysi

STA (Synchronous Time Averaging) eli ajan suhteen synkroninen keskiarvoistaminen toimii periaatteella, jonka mukaan mitattavat suureet keskiarvoistetaan mitattavan kohteen liipaisutietojen perusteella. Koska mittaukset tehdään ajan suhteen synkronoidusti, voidaan mitattavaa suuretta keskiarvoistaa suhteessa rullien pyörimisnopeuteen. Mittaustietojen avulla voidaan piirtää esimerkiksi pyöreysprofiili, jota tulkitsemalla voidaan päätellä mittauskohteen laakerien kunto tai mahdollinen tasapainotuksen tarve. [1] [2]



Kuva 2. STA-analyysin tuloksena muodostettu pyöreysprofiili

Laitteen mahdollinen tasapainotusongelma näkyy pyöreysprofiilissa ympyrän keskiön siirtymisenä pois origosta, kun taas jokin pyörimistä haittaava vika, kuten laakeririkko näkyy piikkeinä ja rosoisuutena, kuten kuvasta 2 on nähtävissä. Pyöreysprofiilia analysoimalla voidaan määrittää, tarvitseeko laite mahdollisesti toimenpiteitä, kuten laakerin vaihtoa, tasapainottamista tai muuta huoltoa. Kasvattamalla mittauksen näytteenottotaajuutta voidaan kasvattaa mitattujen keskiarvojen määrää kierrosta kohden ja jakaa näin pyöreysprofiili pienempiin sektoreihin.

Laajoissa STA-mittauksissa voidaan suorittaa mittauslähteiden suodattimista mittaustuloksista. Usean lähteen mittaustulosten keskinäisestä analysoinnista hyvänä esimerkkinä voidaan pitää useilla puhaltimilla varustettuja jäähdytysjärjestelmiä, joissa monta puhallinta pyörii samalla kierrosalueella. Puhaltimen joukossa on yksi puhallin, joka ei ole enää tasapainossa ilman STA-mittausta jokainen puhallin jouduttaisiin irrottamaan ja tarkastamaan erikseen. Suorittamalla järjestelmän kattava STA-mittaus voidaan säästyä turhilta tasapainotusoperaatioilta sekä minimoida mahdollinen tuotantoon vaikuttava seisokkiaika. [1] [2]

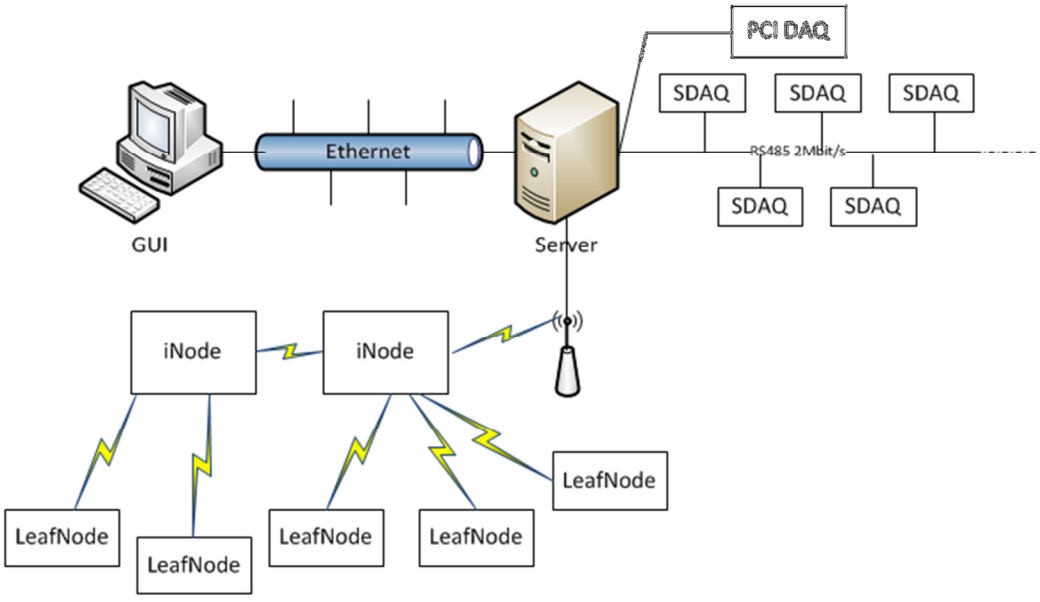
3 WIRELESS MACHINE SENTINEL

WiMS (Wireless Machine Sentinel) on langaton laajennus Machine Sentinel -järjestelmään. Tällä hetkellä WiMS on vielä täysin oma ratkaisunsa, sen liittäminen osaksi tavallista Machine Sentinel järjestelmää vaatii vielä työtä.

Wireless Machine Sentinel koostuu kolmenlaisista komponenteista:

- LeafNodet
  - ovat verkon mittauksista vastaavia asemia
- INodet
  - toimivat tukiasemina LeafNodeille
  - muodostavat keskenään WLAN MESH -verkon, jonka avulla tieto voidaan siirtää tukiasemien välillä langattomasti.
- Palvelin
  - vastaanottaa mittaustiedon ja tekee tarvittavat jälkikäsittelyt.

Alla olevassa kuvassa 3 on nähtävillä, kuinka WiMS on liitetty osaksi Machine Sentinel -järjestelmää.



Kuva 3. Wireless Machine Sentinel osana Machine Sentinel - järjestelmää

3.1 LeafNode

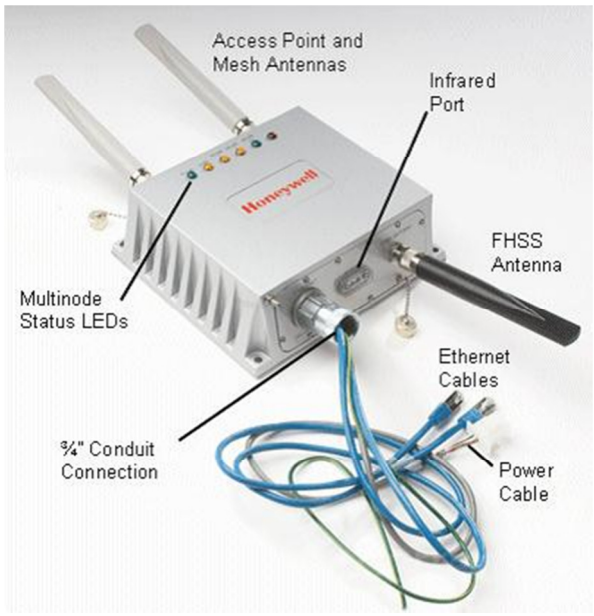
LeafNodet ovat WiMS-verkon rakennepuun haaroja, joiden tehtävänä on mitata paperikoneen laatusignaaleja ja toimittaa tämän jälkeen mittausdata INodejen kautta analysoitavaksi palvelimelle. LeafNodea käsitellään muita järjestelmän osia yksityiskohtaisemmin luvussa 4.

Kuva 4. LeafNode

Yllä olevasta kuvasta 4 on nähtävissä LeafNoden ulkonäkö. LeafNodesta on nähtävillä liittimet, joihin mittaavat sensorit kytketään käyttäen kaapeleita. Tämä mahdollistaa siis leafNoden sijoittamisen turvalliseen paikkaan suojaan iskuilta, kosteudelta ja muilta käyttöikää lyhentäviltä olosuhteilta.

3.2 INode

INode on tukiasema, jonka kautta LeafNodet liittyvät WiMS-verkoon. INode on FIPS-standardit (Federal Information Processing Standards) täyttävä teollisuus-WLAN-tukiasema, joka on suunniteltu toimimaan fyysisesti huomattavasti kovemmissa olosuhteissa kuin tavalliset WLAN-tukiasemat. WLAN-tukiaseman lisäksi Inoden sisällä on OneWireless-tukiasema, jonka avulla verkkoon voidaan liittää Honeywellin OneWireless-tekniikkaa käyttäviä antureita.



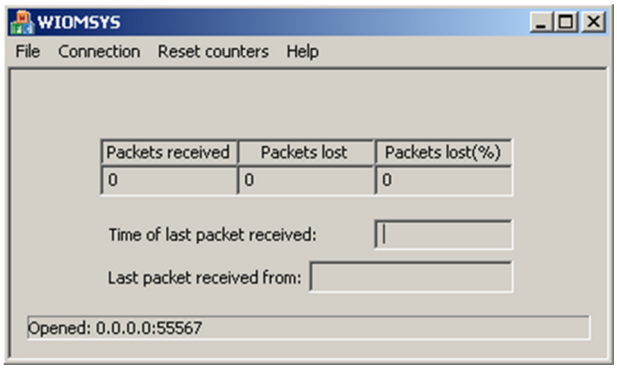
Kuva 5. Inode

INodet voidaan määritellä muodostamaan keskenään WLAN MESH -verkon, jolloin asemien sijoituspaikoille saadaan lisää joustavuutta ja verkon tiedonsiirrolle lisävarmuutta. INodet pystyvät toimimaan samanaikaisesti kahdella eri taajuudella, joten tukiasemien väliset yhteydet voidaan hoitaa käyttäen 802.11a-standardin 5 GHz:n taajuusaluetta. INodeissa on myös PowerOverEthernet-valmius, eli tukiaseman sähköistäminen onnistuu tarvittaessa yhdellä kaapelilla. Kuten kuvasta 5 on nähtävissä, koteloinnissa on panostettu säänkestävyyteen. Vankka valumetallirunko suojaa asemaa fyysisiltä iskuilta, kosteudelta ja pölyltä.

3.3 Palvelin

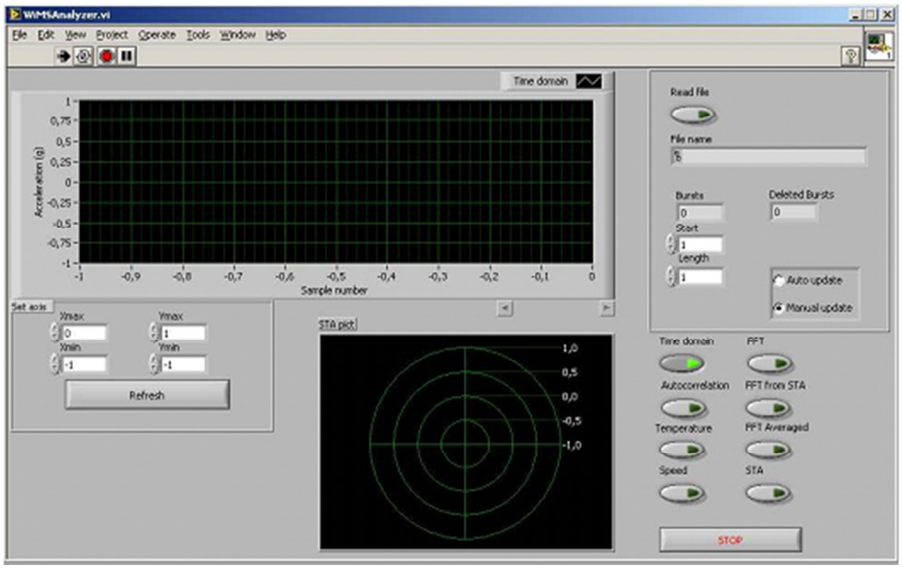
Palvelimella toimii kaksi sovellusta joista WIOMSYS, hoitaa datan vastaanoton liittyvät toimenpiteet ja WimsAnalyzer hoitaa tehtävät laskutoimitukset ja mittaustulosten graafisen esittämisen.

WIOMSYS on palvelimen tiedonkeruusovellus. Se on toteutettu käyttäen C++ olio-ohjelmointikieltä ja Microsoft Visual Studion MFC -luokkakirjastoa. Kuvasta 6 on nähtävissä sovelluksen käyttöliittymä.



Kuva 6. WIOMSYS-käyttöliittymä

WimsAnalyzer on palvelimen tulosten esittämisestä vastaava käyttöliittymäosa. Käyttöliittymän lisäksi WimsAnalyzer hoitaa kaikki datan analysointiin liittyvät laskutoimitukset. Sovellus tarjoaa paperikoneen operaattorille STA-kuvaajan lisäksi ajonopeudet, FFT- kuvat ja kiihtyvyysdatan ajan suhteen. Kuvasta 7 on nähtävissä WimsAnalyzer-sovelluksen käyttöliittymä.



Kuva 7. WimsAnalyzer-käyttöliittymä

4 LEAFNODE

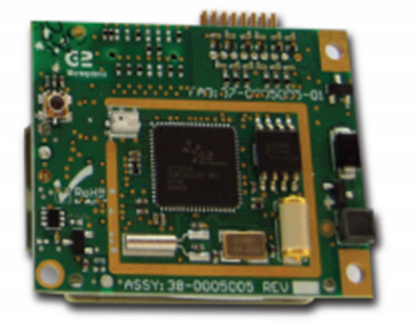
LeafNode on Wireless Machine Sentinelin mittauksista vastaava osa. Mittauskortti, radiokortti ja laitteen paristo sijaitsevat koteloinnin sisällä. Pulssianturi ja laatusignaalianturi ovat sijoitettavassa erilleen leafNodesta.

Kuva 8. LeafNoden osat

LeafNode voidaan jakaa kuvan 8 mukaisesti kolmeen mittauksen kannalta olennaiseen osaan: antureihin, mittauskorttiin ja radiokorttiin.

4.1 Radiokortti C501RT01

Tämänhetkisessä kehitysversiossa käytetään G2 Microsystemsin C501RT01-referenssikorttia, jolla on mahdollista testata kaikkia G2C501-piirin ominaisuuksia.
Koska kortti on suunnattu WLAN-seurantaan, on kortin lähes kaikki käytettävissä olevat yleiskäyttöiset liitännät jo käytössä sensoreilla ja lisälaitteilla. Kortilla on ainoastaan kaksi pinniä, jotka ovat täysin vapaana ilman, että samassa linjassa olisi jokin sensori tai komponentti. Kuvasta 9 on nähtävillä koteloimaton C501RT01-kortti.



Kuva 9. G2C501RT01-kortti

Radiokortti sisältää seuraavat komponentit:

- täydellinen 2.4GHz:n IEEE 802.11b PHY/MAC
- ISO 24730 Reaaliaikapaikannuslähetin
- 900MHz:n EPC RFID
- 125kHz RFID-rajapinta
- erillinen AES-salain
- täysin ohjelmoitavissa oleva 32-bittinen Leon 2 Sparc V8 suoritin, jossa on rajapinta ulkoiselle flash-muistille, jolle voidaan tallettaa esimerkiksi suoritettavat ohjelmat.
- 64Kt RAM-muistia sekä 2Kt paristo varmennettua RAM-muistia.
- 384Kt ROM muistia, joka sisältää eCos-käyttöjärjestelmän, TCP/IP-pinon ja laiteajurit.
- Reaaliaikakello [3]

Kortille sijoitettujen komponenttien avulla C501RT01 on helppo valjastaa esimerkiksi laajoihin varastointijärjestelmiin kertomaan tuotteen sijaintia, lämpötilaa, tuotteen saamia iskuja, mahdollisten elektronisten sinettien murtumista tai muuta vastaavaa.

4.1.1 G2C501 SoC

LeafNoden radiokortin toiminnasta vastaa G2 Microsystemsin G2C501 SoC-piiri (System on a Chip). Vähäisen virrankulutuksensa vuoksi G2C501 sopii erityisesti paristokäyttöisiin laitteisiin. Piirin suorittimena toimii 32-bittinen Leon 2 Sparc V8-prosessori, joka on Aeroflex Gaislerin Euroopan avaruusjärjestölle suunnittelema avoimen lähdekoodin VHDL-toteutus. Lisäksi piirillä on laitteistotason toteutukset C501RT01:n komponenteille, sekä SPI-väylälle ja UART:lle. [3]

Johtuen G2C501-piirin sisäisen flash-muistin puutteesta, joudutaan muistin kytkemiseksi piiriin käyttämään SPI-väylää. Väylälle laitetta lisättäessä on kuitenkin huolehdittava, ettei mikään väylän laitteista pääse sotkemaan sovellusohjelman lataamista ulkoisesta flash-muistista. Mikäli väylällä on käynnistyksen yhteydessä flash-muistin lisäksi jokin muu laite, voi seurauksena olla muistin sisällön korruptoituminen tai sovelluksen käynnistymisen epäonnistuminen.

4.1.2 eCos

eCos on avoimen lähdekoodin reaaliaikakäyttöjärjestelmä, joka on suunnattu erityisesti sulautettuihin järjestelmiin. Vaikka eCos on Linux-käyttöjärjestelmästään tutun Red Hatin ylläpitämä, ei se kuitenkaan liity millään tavalla Linuxiin, vaan on täysin oma projektinsa. [4]

eCos on suunniteltu toimimaan monenlaisilla eri arkkitehtuureilla ja alustoilla, tällä hetkellä eCos tukee 13 eri arkkitehtuuria. eCos toimii 16, 32 ja 64-bittisillä arkkitehtuureilla, mikroprosessoreista mikrokontrollereihin ja signaaliprosessoreihin. eCos ydin, sen kirjastot ja ajettavat komponentit on kerrostettu Hardware Abstraction Layerille, joten sen saa toimimaan lähes missä tahansa, kunhan HAL ja tarvittavat laiteajurit ovat sovitettu kohdearkkitehtuurille. [4]

Muokattavuutensa johdosta eCos sopii myös erittäin rajallisin resurssein varustettuihin toteutuksiin. eCos julkaistaan muokatun GPL-lisenssin alaisena ja näin sitä siis voidaan käyttää täysin ilmaiseksi kaikkialla. Lisenssin avulla rajoitetaan ainoastaan eCos-koodin väärinkäyttöä, kuten käytetyn koodin patentoimista tai muuta vastaavaa, josta voisi olla haittaa koko eCos-yhteisölle. Lisenssi ei pakota käyttäjää julkaisemaan eCosin päälle kirjoitettua ohjelmakoodia, ainoastaan eCos-lisenssin alla olevan koodin muokkaukset täytyy julkaista. G2 Microsystemsin osalta eCos toimii käyttöjärjestelmän pohjana tarjoamassa reaaliaikakäyttöjärjestelmän tärkeimpiä ominaisuuksia, kuten moniajoa ja muistinhallintaa. [4]

eCos tärkeimmät ominaisuudet ovat:

- Hardware Abstraction Layer (HAL)
- reaaliaikaydin
- keskeytysten ja poikkeusten hallinta
- ajastimen valinta
- säikeistäminen
- synkronisointi primitiivien suuri määrä
- ajastimet, laskurit ja hälytykset
- debuggaus ja instrumentointituki
- µITRON 3.0 yhteensopiva API
- POSIX yhteensopiva API
- ISO C ja matematiikka-kirjastot
- sarjaliikenne, ethernet, SPI, I2C, framebuffer, CAN, ADC, watchdog-laiteajurit

- USB-slave tuki
- TCP/IP-liikennöintipino
- C++ Stardard Template Library
- GDB Debug-tuki [4]

G2 Microsystemsin käyttämä eCos on valmistajan itse muokkaama versio, joten siinä on ainoastaan valmistajan tarvitsemakseen katsomat osat.

4.2 Mittauskortti

Mittauskortin toiminnoista vastaan MSP430F1611-mikro-ohjain, joka yhdessä kontrollerilla pyörivän µC/OS-II – reaaliaikakäyttöjärjestelmän, tarjoaa reaaliaikaista mittaustietoa liitetyiltä sensoreilta. Mittausten valmistuttua mittauskortti toimittaa mittausdatan SPI-väylään liitetylle radiokortille, jonka tehtävä on toimittaa data eteenpäin.

4.2.1 Texas Instruments MSP430F1611

MSP430F1611 on erittäin vähävirtainen 16-bittinen MSP430-mikro-ohjain, joka sopii erittäin hyvin kohteisiin missä halutaan päästä mahdollisimman pieneen virrankulutukseen. Standby-tilassa MSP430F1611 kuluttaa ainoastaan 1,1 µA ja off-tilassa ainoastaan 0,2 µA, joka kuluu RAM-muistin ylläpitoon. Reaaliaikaisuus vaatimuksien kannalta sen hyviä ominaisuuksia ovat 125 ns käskytysväli, sekä 6 µs heräämisaika, lisäksi MSP430-arkkitehtuuriin perustuvat prosessorit ovat monien reaaliaikakäyttöjärjestelmävalmistajien tukemia, joten reaaliaika-käyttöjärjestelmän käyttöönotto ei vaadi tavallisesti kovinkaan suurta työtä. Juuri tämän vuoksi MSP430F1611-mikro-ohjain sopii erityisesti kannettaviin ja paristokäyttöisiin mittalaitteisiin. [5]

4.2.2 µC/OS-II

µC/OS-II on täysin siirrettävä, skaalautuva, reaaliaikainen moniajoon pystyvä ydin, joka on sovitettu jo yli neljällekymmenelle prosessoriarkkitehtuurille aina 8-bittisistä 64-bittisiin prosessoreihin. µC/OS-II:a käytetään ympäri maailmaa monenlaisissa erilaisissa sovelluksissa, kuten esimerkiksi kameroissa, lääketieteellisissä instrumenteissa, moottorin ohjauksissa ja teollisuusroboteissa. [6]

µC/OS-II soveltuu erityisen hyvin turvallisuuskriittisiin kohteisiin, kuten lentokoneisiin, lääketieteellisiin laitteisiin, ydinvoimajärjestelmiin ja liikenteeseen, johtuen sen saa-

mista esihyväksynnöistä ja sertifikaateista. µC/OS-II:n jakelusta vastaa Micrium Corporation, jonka johdossa on edelleen yrityksen perustaja Jean Labrosse. [7]

4.3 Anturit

LeafNodeen kytketään kahdentyyppisiä antureita:

- Pulssiantureita joiden avulla saadaan selville paperikoneen pyörivien osien kierrosnopeudet.
- Laatusignaaliantureita joiden avulla saadaan laatutietoa prosessista. Laatusignaalianturit voivat olla joko kiihtyvyysantureita tai paineantureita.

Alla olevasta kuvasta 10 on nähtävissä käytettävät kiihtyvyys- ja pulssianturit.

Kuva 10. Pulssi- ja kiihtyvyysanturi

Pulssianturi antaa jokaisella kierroksella liipaisupulssin, jonka perusteella voidaan määrittää esimerkiksi rullan pyörimisnopeus. Liipaisupulssit toimivat myös perustana mittauksista tehtäville taajuusanalyyseille.

4.4 Adapteri

Adapteriksi kutsuttu kortti toimii välikappaleena mittauskortin ja radiokortin välissä. Adapterikorttia käytetään liitäntöjen tuomiseksi radiokortin saataville, ja se toimii myös SPI-väylän erottajana. Adapterilevylle on sijoitettu 74AHC126-puskuripiiri, jonka avulla mittauskortin ja radiokortin välinen yhteys voidaan saattaa korkeaimpedanssitilaan ja näin mahdollistaa turvallinen tiedon lukeminen samalla väylällä sijaitsevalta flash-muistilta. Puskuripiirin avulla varmistetaan esimerkiksi G2C501:n sovellusohjelman lataamisen onnistuminen käynnistyksen yhteydessä.

4.5 Käyttövoima

Mittausaseman tehonlähteenä toimii 3,6 V:n D-koon litiumparisto. Käyttöjännite otetaan suoraan paristolta. Sekä mittauskortilla että radiokortilla on oma virransyöttöelektroniikkansa.

5 VERKKOTEKNIIKAT

5.1 OneWireless

OneWireless on Honeywellin tarjoama langaton verkkoratkaisu, joka on suunnattu tehtaiden kriittisten prosessiparametrien tarkkailuun. OneWireless-laitteet toimivat 900 MHz ja 2,4 GHz:n taajuusalueilla.

OneWireless-verkkoratkaisua markkinoidaan kolmen s-kirjaimen avulla: [8]

- Simple
    - Verkon käyttöönotosta ja ylläpidosta on tehty helppoa.
- Safe
    - OneWireless käyttää ainoastaan todistettua tekniikkaa
    - Sijoitukselle luvataan suojaa tarjoamalla alaspäin yhteensopivaa teknologiaa. Näin helpotetaan tulevaisuudessa siirtymistä uuteen teknologiaan.
- Secure
    - OneWireless-verkko tarjoaa turvallisuutta koko järjestelmän kattavasti ominaisuuksilla, kuten laitteiden tunnistus, AES-salaus ja viestien reitityksen hallinta.
    - Lähettimet käyttävät FHSS-taajuushyppelyteknologiaa, viestien sarjanumeroita ja 128-bittisiä salausavaimia sekä 64-bittisiä viestien yhtenäisyyskoodeja.

OneWireless on suunniteltu vaihtoehdoksi kohteisiin, joissa kaapeloinnin tekeminen on vaikeaa tai mahdotonta.

5.2 IEEE802.11

IEEE802.11-standardit määrittelevät WLAN-verkon MAC-toiminnat ja fyysisen kerroksen. Alkuperäinen IEEE 802.11-standardi määrittelee 1 Mb/s ja 2 Mb/s tiedonsiirtonopeudet käyttäen DSSS- ja FHSS-tekniikkaa. Radiotien lisäksi standardi määrittelee infrapunasiirtotien. Standardiin liittyvät laajennukset merkitään aakkosin päästandardin perään. Tunnetuimpia laajennuksia ovat varmasti siirtonopeuteen vaikuttavat a, b, g ja n -laajennukset.

5.2.1 IEEE802.11-standardin laajennukset

802.11-standardiin on määritelty seuraavat siirtonopeuteen vaikuttavat laajennukset

- IEEE802.11a
    - Määrittelee fyysisen kerroksen toteutuksen OFDM-modulointia käyttäen 5 GHz:n taajuusalueella ja määrittelee päästandardiin vaadittavat lisäykset [9].
- IEEE802.11b
    - Määrittelee, kuinka High Rate -nopeuslaajennus toteutetaan fyysisellä tasolla käyttäen DSSS-modulointia. High Rate -laajennus mahdollistaa maksimissaan 11 Mb/s -nopeuden 2,4 GHz:n ISM-taajuusalueella. [10]
- IEEE802.11g
    - Määrittelee, kuinka toteutetaan 54 Mb/s nopeuslaajennus 2,4 GHz:n taajuusalueelle [11].
- IEEE802.11n
    - Määrittelee, kuinka teoreettinen 600 Mb/s tiedonsiirtokapasiteetti voidaan saavuttaa [12].

Lisäksi IEEE802.11-standardilla on useita muita vähemmän tunnettuja laajennuksia, joilla parannetaan verkon turvallisuutta, virransäästöominaisuuksia, laitteiden yhteensopivuutta eri maiden lainsäädäntöjen kanssa. Näihin standardeihin kuuluu myös IEEE802.11s, joka määrittelee WLAN MESH -verkkojen toteuttamisen, tosin se on vielä keskeneräinen standardi. [13]

5.2.2 IEEE802.11 Beacon-kehys

Beacon-kehys on IEEE802.11-aseman lähettämä verkonhallintakehys, jonka avulla kerrotaan mahdollisille verkkoon liittyjille verkon olemassaolosta, ja ilmoitetaan samalla yhteyden käyttöön liittyvistä vaatimuksista tai mahdollisuuksista. [13]

Beacon-kehyksen sisältö riippuu käytetyn aseman tukemista standardeista, mutta tavallisesti beacon-kehyksessä lähetetään ainakin seuraavat tiedot:

- Timestamp
  - aikaleima, joka annetaan kehykselle lähetyksen yhteydessä.
- Beacon-interval
  - kertoo beacon-kehyksen lähettämisvälin.
  - tärkeä tieto esimerkiksi virransäästöä käyttäville asemille
  - tavallisesti lähetysväli on 100 millisekunnin luokkaa.
- Capability
  - tämä kenttä kertoo esimerkiksi, onko käytössä infrastruktuuri- vai Ad-Hoc-verkko, tai mitä salausta verkossa käytetään.
- SSID (Service Set Identifier)
  - verkon tunniste, jonka avulla verkkoon liittyvä asema voidaan tunnistaa
- Supported rates
  - kertoo beacon-kehyksen lähettäjän tukemat siirtonopeudet
- Parameter Sets
  - FH-, DS-, CF- ja IBSS-parametrit, jotka tarvitaan yhteyden käyttämiseen
  - parametrit riippuvat käytössä olevasta tiedonsiirtotavasta, eli mikäli käytössä on esimerkiksi taajuushyppely ilmoitetaan siihen liittyvät parametrit.
- Traffic Indication Map (TIM)
  - kertoo virransäästöä käyttäville asemille jos niille on datakehyksiä tukiaseman puskurissa. [13]

Lisäksi Beacon-kehys voi sisältää tietoja, kuten maatiedon tai lähetysvoimakkuudenrajoitteet, nämä tiedot välitetään IEEE802.11-2007-standardin mukaan tehdyissä laitteissa. [13]

5.2.3 IEEE802.11 TSF

TSF (Timing synchronization function) on IEEE802.11-standardin mukainen kellojen synkronointitapa. TSF määrittelee, kuinka kellot synkronoidaan yhden BSS:n (Basic Service Set) alaisuudessa.

5.2.3.1 Infrastruktuuriverkot

Infrastruktuuriverkoissa verkon tukiasema toimii ajastuksen isäntänä ja sen laskuri toimii täysin itsenäisesti riippumattomana muista verkon tukiasemista. Tukiasema lähettää tietyin väliajoin Beacon-kehyksen, joka sisältää tiedon tukiaseman TSF-laskurin ajasta. Infrastruktuuriverkoissa Beaconin vastaanottanut asema muokkaa aina oman TSF-laskurinsa tukiaseman TSF-laskurin aikaan. [13]

5.2.3.2 AdHoc-verkot

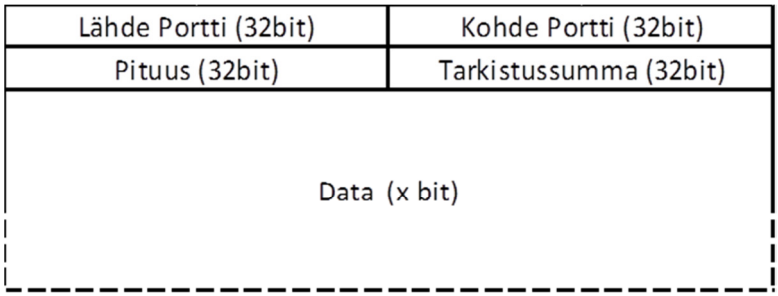
AdHoc-verkoissa synkronointi tapahtuu hieman eri tavalla kuin infrastruktuuriverkoissa. Synkronoidakseen kellonsa TBTT:n (Target Beacon Transmission Time) jokaista TBTT:tä kohden asemien tulee käyttää seuraavaa algoritmia: [13]

1. Keskeytetään back-off-ajastimen vähentäminen kaikilta vireillä olevilta ei-beacon ja ei-adHoc-liikenteenindikointilähetyksiltä.
2. Lasketaan satunnainen viive, joka on väliltä 0 - 2×aCWmin×aSlotTime.
    - aCWmin ja aSlotTime ovat vakioita, jotka riippuvat käytetyn siirtotien moduloinnista.
3. Odotetaan satunnainen viive samalla vähentäen satunnaisviivelaskuria samalla algoritmilla kuin back-off-laskuria.
4. Peruutetaan jäljelle jäävä satunnainen viive ja vireillä ollut beacon-kehyksen lähetys, jos kehys saapuu ennen satunnaisviivelaskurin vanhenemista.
5. Lähetetään beacon-kehys, jos satunnainen viive on kulunut loppuun eikä beacon-kehystä ole vastaanotettu muilta verkon asemilta.

Vastaanotettuaan beacon-kehyksen asema tarkastaa mukana toimitettu TSF-laskurin aikaleiman ja mikäli kehyksessä toimitettu aikaleima on suurempi kuin paikallisen TSF-laskurin arvo, päivittää asema paikallisen TSF-laskurin vastaanotetulla laskurin arvolla. Tällä tavalla varmistetaan, että nopeimmin käyvä kello on aina johdossa ja näin vältytään kellon siirroilta taaksepäin. [13]

5.3 UDP-protokolla

UDP (User Datagram Protocol) on yksinkertainen, epäluotettava ja yhteydetön protokolla. UDP sopii erityisesti sovelluksiin joissa lähetetty tieto vanhentuu nopeasti ja pienet datahäviöt siedetään. Kuten kuvasta 11 näkyy, UDP-kehys sisältää ainoastaan tiedot lähde- ja kohdeportista, tarkistussumman sekä kehyksen pituuden, näitä tietoja seuraa datakenttä. [14]



Kuva 11. UDP-kehyksen rakenne

UDP-protokollan epäluotettavuus johtuu sen uudelleenlähetysmekanismin puutteesta, uudelleenlähetys on siis protokollan käyttäjän vastuulla. Yhteydettömyytensä ansiosta UDP-protokolla toimii pohjana monille anycast- ja broadcast-lähettämistä vaativille protokollille. Merkittävin UDP-protokollan käyttökohde on verkonhallinta. [14]

Yhteydettömyytensä vuoksi UDP-protokolla sopii hyvin reaaliaikaiseen tiedonsiirtoon, koska tiedonsiirron aloittaminen ei vaadi erillistä yhteyden avaamista. Yhteydettömyydessä on kuitenkin omat ongelmansa, esimerkiksi jos lähetettävässä tiedossa ei haluta olevan häviöitä, joudutaan keskittymään tarkemmin tiedon mahdolliseen uudelleenlähettämiseen tai muuten parantamaan järjestelmän toipumista mahdollisista menetetyistä paketeista. [14]

6 AIKASYNKRONOINTI JA AIKALEIMAUS

6.1 Miksi aikasynkronointia tarvitaan?

Aikasynkronointia tarvitaan yhtäaikaisien tapahtumien mittaamiseen ja toteuttamiseen. Mittaamisen kannalta hyvä esimerkki on akustinen paikannus ja toimintojen kannalta esimerkiksi energiapihi radiokanavan käyttö.

Synkronointivaatimukset kasvavat tekniikan kehittyessä. Reaaliaikaisen mittausjärjestelmän synkronoinnin tarkkuusvaatimukset voidaan tavallisesti johtaa mittausjärjestelmän näytteenottotaajuudesta, eli näytteenottotaajuuden kasvaessa kasvaa myös aikasynkronoinnin tarkkuusvaatimus.

Wireless Machine Sentinelissä aikasynkronointia tarvitaan, koska jokaisella verkon asemalla on oma kellonsa ja jokainen kelloista käy luonnollisesti hieman eri tahtiin. Yhden millisekunnin tarkkuudella tarkoitetaan, että jokaisen verkonmittausaseman tulee pystyä ilmoittamaan tiettynä ajankohtana tapahtunut mittaus siten, että jokaiselta asemalta saatu mittaustulos saadaan 1 ms kokoisen ikkunan sisään.

6.2 Aikasynkronoinnin vaatimukset ja tarpeet

Ajan ylläpitämiseen langattomissa ratkaisuissa vaikuttavat monet asiat, kuten kellon stabiilius, käytössä oleva synkronointikanava ja synkronoinnin toteutustapa.

Synkronoinnin tarpeet voidaan jaotella seuraavilla tavoilla: [15]

- Globaali vai paikallinen synkronointi
    - Täytyykö mittaustulokset saada samalle viivalle toisella puolella maailmaa tehtyjen mittausten kanssa vai riittääkö, että mittaustuloksia voidaan vertailla keskenään paikallisen verkon asemien kanssa?
- Absoluuttinen vai suhteellinen aika
    - Tarvitaanko mittauksista absoluuttinen aikaleima vai riittääkö tieto mittausten aikaerosta toisiinsa?
- Hardware- vai Software-toteutus

    - Käytetäänkö synkronoinnin apuna siihen täysin suunnattua hardware-toteutusta vai käytetäänkö sovellustason synkronointia, eli synkronointi hoidetaan muun dataliikenteen seassa?
- Apriorinen vai postapriorinen
    - Pidetäänkö kellonaikaa koko ajan yllä vai suoritetaanko synkronointi tarpeen vaatiessa?
- Deterministinen vai stokastinen tarkkuus
    - Voidaanko synkronoinnin tarkkuudelle antaa jokin tietty arvo, vai pelkkä todennäköisyys?

Wireless Machine Sentinelissä synkronointiin riittää paikallinen paperikoneen kattava synkronointi. Aikaleimauksen täytyy olla absoluutinen, jotta mittaustulokset voidaan piirtää samalle aikajanalle ja toteutus tulisi olla software-pohjainen eli käyttää synkronointiin käytössä olevaa WLAN-rajapintaa.

Aikasynkronointitapoja voidaan vertailla keskenään seuraavilla parametreilla: [15]

- Tarkkuus
    - tarkkuus noden ja oikean ajan välillä
    - tarkkuus suhteessa muihin verkon nodeihin
- Energiankulutus
    - synkronointitiheys
    - synkronointiviestien määrä
- Muistinkäyttö
    - protokollan vaatima käyttömuistin tarve
- Virheistä toipuminen
    - toipumisnopeus
    - toipumistapa

6.3 Synkronointiin vaikuttavat tekijät

Synkronoinnin tarkkuuteen vaikuttavat monet tekijät, kuten käytettävien kellojen laatu, synkronointi prosessi ja synkronointi prosessiin liittyvien toimintojen tarkkuus.

6.3.1.1 Kellon laatu

Kellon laadulla voidaan vaikuttaa erittäin paljon synkronointiprotokollan vaatimuksiin. Huonolaatuinen kello kasvattaa synkronointiprotokollan ja synkronoitiin käytetyn kanavan vaatimuksia.

Kellon laatu määritellään sen käymistaajuuden stabiiliudella. Kellonsignaalin luomiseen käytetään aina jotain fyysistä prosessia, kuten heilurin liikettä, kiteen värähtely taajuutta tai kuten atomikellossa atomien resonanssitaajuudesta. Kellon laatuun vaikuttaa myös sen käymistaajuuden tarkkuus, mutta käymistaajuuden virheet voidaan korjata kalibroinnin avulla. [16]

6.3.1.2 Synkronointiprosessi

Synkronointiprosessi voi perustua vaihe-eron tai taajuuseron korjaamiseen tai näiden yhdistelmään. Vaihe-ero kertoo, kuinka kaukana kellot ovat toisistaan tietyllä ajanhetkellä. Taajuusero kertoo eron kellojen käyntitahdissa eli sen kuinka paljon kellojen poikkeama muuttuu ajanfunktiona. Synkronointiprosessin kannalta suurimpia virheenaiheuttajia ovat satunnaiset viiveet, jotka tulisi kartoittaa mahdollisimman tarkasti niiden vaikutus minimoimiseksi. [16]

Langattomien sensoriverkkojen synkronointiprosessiin kuluva aika voidaan jakaa seuraavasti: [16]

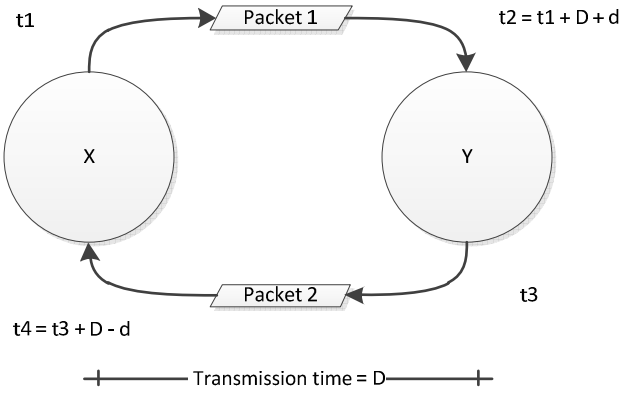
- Send time
    - lähettämiseen kuluva aika (puskurointiin, käsittelyyn tms.)
    - riippuu prosessorin suorituskyvystä ja muista resursseista
- Access time
    - aika joka kuluu odotellessa lähetyskanavaa
    - riippuu kanavan käytöstä
- Propacation time
    - aika joka kuluu signaalin siirtymiseen kanavalla
    - riippuu lähettäjän ja vastaanottajan välisestä etäisyydestä.
- Receive time
    - aika joka kuluu, ennen kuin vastaanotettu data on valmis käsiteltäväksi (aikaleimattavaksi).
    - riippuu viestin pituudesta.

6.4 Kelloparin synkronointi

Monet langattomien sensoriverkkojen protokollat perustuvat kelloparien välisiin synkronointeihin

6.4.1 Pareittain synkronointi

Pareittain synkronointi on symmetrisille yhteyksille sopiva yksinkertainen kellojen synkronointitapa, jolla kellojen päivittäminen onnistuu kolmen viestin avulla. Symmetrisen yhteyden lisäksi täytyy lähetettävien synkronointipakettien olla kooltaan symmetrisiä.



Kuva 12. Pareittain synkronointi

Synkronointi tapahtuu seuraavasti:

1. Node X lähettää oman paikallisen kellonaikansa t_1 nodelle Y.
2. Node Y tallentaa vastaanottohetken omana kellonaikanaan t_2.
3. Node Y lähettää aikaleiman t_3 nodelle X (paketti sisältää myös t_1 ja t_2 leiman).
4. Node X tallentaa vastaanottohetken omana kellonaikanaan t_4.
5. Node X laskee kellojen välisen poikkeama d ja lähettää lopuksi mahdollisen kellonmuokkausviestin nodelle X.

Kellon poikkeama d ja siirtoon kuluva aika D voidaan laskea kaavojen 1 ja 2 avulla.

$$d = \frac{(t_2 - t_1) + (t_4 - t_3)}{2} \quad (1)$$

$$D = (t4 - t1) - (t3 - t2) \quad (2)$$

6.4.2 TinySync ja MiniSync

TinySync ja MiniSync ovat samanlaisia kuin perinteinen pareittain synkronointi, erolla että kelloja ei kuitenkaan synkronoida, vaan kellojen parametreista muodostetaan historiatietoa. Näiden perusteella voidaan laskea kellojen parametrit, joiden perusteella voidaan tulkata kellon X aika kellon Y ajaksi. [15]

6.5 Lähettäjän ja vastaanottajan välinen synkronointi

Tavallisin tapa lähestyä kellonsynkronointia on paikallisen kellon muokkaaminen luotettavan ajanlähteen mukaan. Tästä hyviä esimerkkejä ovat yleisesti käytetty NTP-protokolla sekä sensoriverkkoihin suunnattu IEEE1588 PTP. Tavallisimmat synkronointiprotokollat pyrkivät synkronoimaan kelloa siten, että kellolta saatu aikaleima on aina paikkansa pitävä ja lopullinen.

6.5.1 LTS

LTS-protokollassa (Lightweight Time Synchronization) pyritään jakamaan verkon luotettavimmilla ajanlähteillä varustetun noden kellonaikatieto muille verkon nodeille suorittaen pareittaissynkronointeja naapuriasemien välillä. Synkronoinnit tehdään puurakennemaisesti. Puurakenteen selvittämiseen voidaan käyttää esimerkiksi DDFS-algoritmia (Distributed Depth-First Search) tai kaiuttamisalgoritmeja. [15]

LTS-protokolla voi olla keskitetty tai jaettu, keskitetyssä versiossa synkronoinnin juurena toimii yksi referenssinode, jonka kellonlähde on muita tarkempi, esimerkiksi GPS-vastaanotin. Ensin referenssinode suorittaa pareittain synkronoinnin sen alaisten kanssa, minkä jälkeen alaiset suorittavat aikasynkronoinnin alaistensa kanssa ja niin edelleen, kunnes kaikki verkon nodet on synkronoitu. Jaetussa versiossa ei luoda puurakennetta, mutta jokainen node tietää reitin lähimmälle referessinodelle, ja näin ne voivat pyytää synkronointia lähimmältä referenssinodelta. [15]

6.5.2 TPSN

TPSN-protokollassa (Time-sync Protocol for Sensor Network) synkronointi tehdään keskitetyn LTS-protokollan tapaan hierarkkisesti muodostaen puurakenne käyttäen juuri-nodea rakenteen juurena. Synkronointirakenteen muodostamiseksi nodet lähettävät niin sanottuja level-discovery -viestejä. Synkronointipuun muodostaminen tapahtuu siten, että juuri-node lähettää ensin level-discovery -viestissä tiedon, jossa se

kertoo sijaitsevansa tasolla nolla, tämän perusteella yhden hypyn päässä juurinodesta sijaitsevat nodet päivittävät omaksi tasokseen tason yksi ja lähettävät omat level-discovery -viestinsä. Vastaanotettuaan level-discovery -viestin node asettaa viestinlähettäjän synkronointi-isännäkseen ja ohittaa kaikki seuraavat level-discovery -viestit. [15]

TPSN-protokollassa aikaleimaaminen on toteutettu MAC-tasolla juuri ennen paketin lähettämisen aloittamista kanavalle, näin päästään eroon käyttöjärjestelmän aiheuttamista viiveistä lähetyspäässä, myös vastaanotettu paketti aikaleimataan erittäin varhain vastaanoton keskeytysrutiinissa. [15]

Synkronoitaessa korkeamman tason node X synkronoi kellonsa noden X-1 kellonaikaan, synkronointi tapahtuu pareittais-synkronoinnilla, jonka tuloksena voidaan laskea kellon poikkeama ja korjata kellonaika. [15]

TPSN-protokollan avulla voidaan saavuttaa lähes 17 µs tarkkuus(1 hyppy), mutta TPSN tarvitsee MAC-tason aikaleimaukset, sekä se sallii mielivaltaiset hypyt noden kellonajassa. [15]

6.5.3 NTP

NTP (Network Time Protocol) on erittäin laajasti käytetty kellojen synkronointiprotokolla, erityisen laaja ja tunnetuin käyttö kohden NTP:lle on internet. Nykyään NTP-protokolla vastaa noin 25 miljoonan tietokoneen kellon synkronoinnista. [17]

Johtuen NTP-protokollan suuntautumisesta lähinnä tietokoneverkkojen synkronointiin, ei sen sopivuus langattomiin sensoriverkkoihin ole kovin hyvä. NTP-protokolla on suunniteltu periaatteella, jossa prosessoriaika ja verkonkäyttö on lähes ilmaista. NTP-protokollan avulla saavutettava tarkkuus on tavallisesti millisekunti-luokka. Protokollaan kuuluu kellonsynkronointi-prosessien lisäksi myös monia turvallisuuteen vaikuttavia tekijöitä, jotka ovat välttämättömiä internetin kaltaisessa kaikille avoimessa verkossa. [17]

NTP-aliverkko toimii hierarkkisissa tasoissa, joita kutsutaan Stratumeiksi.

- Stratum 1 tason palvelimet ovat suoraan yhdistettyjä kansallisiin kellonaika lähteisiin, satelliitti-, radio tai puhelinyhteydellä.
- Stratum 2 tason palvelimet synkronoivat kellonsa Stratum 1 palvelimen tarjoamaan kellonaikaan

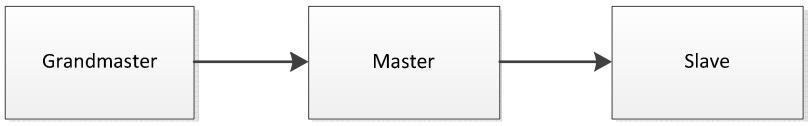
- Stratum 3 tason palvelimet taas synkronoivat kellonsa Stratum 2 palvelinten kelloihin ja niin edelleen. [17]

Aikatiedon vaihtaminen onnistuu samalla tavalla, kuten perinteisessä pareittais-synkronoinnissa, eli asiakas lähettää pyynnön, jonka se aikaleimaa lähetysvaiheessa aikaleimalla t1, palvelin vastaanottaa pyynnön ja leimaa vastaanottohetken t2, tämän jälkeen palvelin lähettää vastauksen, joka leimataan lähetyksen yhteydessä t3 ja lopuksi asiakas vastaanottaa vastauksen ja tallentaa vastaanottohetken t4. Näiden aikaleima tietojen perusteella lasketaan kellonpoikkeama ja viive, joiden perusteella voidaan säätää kellon aikaa ja taajuutta [17].

6.5.4 IEEE1588 PTP

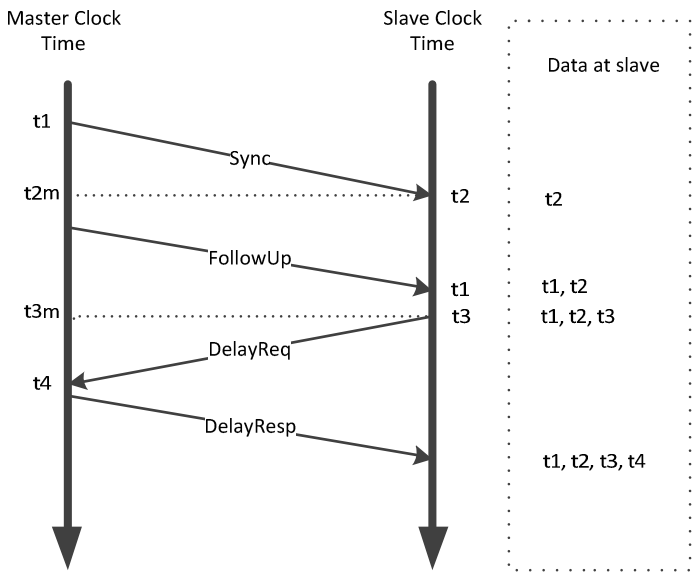
IEEE1588 on suunniteltu täyttämään NTP-protokollalle sopimattomien kohteiden tarpeet. IEEE1588-standardi määrittelee hierarkkisen isäntä-orja arkkitehtuurin kellonajan jakamiseen. Protokolla toimii periaatteella, jossa orja-laitteet synkronoivat oman sisäisenkellonsa isäntä-kellon kanssa, käyttäen tietynlaista synkronointimekanismia. Tulevaisuudessa NTP:n ja IEEE1588 PTP:n välistä yhtyeentoimivuutta tullaan parantamaan. [18]

Kun IEEE1588 laite yhdistetään 1588-topologiaan, sen täytyy lähettää multicast SYNC-viesti. SYNC-viesti sisältää tietoa laitteesta ja kaikista sen sidoksista. Kaikki verkon laitteet vastaanottavat SYNC-viestit ja lähettävät omat SYNC-viestinsä. Tämän jälkeen verkon laitteet suorittavat BMC-algoritmin (Best Master Clock), jonka avulla verkon laitteet muodostavat synkronointisuhteet toisiinsa. Standardissa määritellään algoritmi parhaan isäntä-kellon valinnalle. Parhaan isännän valitsemiseen käytetään tietoja, kuten kellon aikajakauma, prioriteetti ja varianssi. Kun laite on suorittanut BMC-algoritmin ja valinnut laitteen jonka kanssa se synkronoi kellonsa, voidaan aloittaa synkronointimekanismi. [19]



Kuva 13. Synkronointi-suhteet

Synkronointimekanismi on neljän kehyksen sarja, joka lähetetään isäntä- ja orja-laitteen välillä. Kuvasta 14 on nähtävillä synkronointimekanismin viestiliikenne.



Kuva 14. Synkronointimekanismi

1. Ensimmäinen kehys on isäntälaitteen lähettämä Sync-viesti, jonka lähetys hetki tallennetaan, tässä tapauksessa nimellä t_1. Slave vastaanottaa Sync-viestin, ja tallentaa kehyksen vastaanottohetken t_2.
2. Sync-viestin jälkeen isäntälaite lähettää Follow_Up – viestikehyksen, joka sisältää Isäntälaitteelta lähetetyn Sync-viestin lähetysajankohdan.
3. Orjalaite vastaa tähän Delay Request-viestillä ja tallentaa Delay_Request-viestin lähetysajankohdan t_3.
4. Kun Isäntä vastaanottaa Delay_Request -viestin, se vastaa Delay_Response-viestillä, joka sisältää Delay_Request -viestin vastaanottohetken t_4.

Synkronointimekanismin avulla saatujen aikaleimojen perusteella voidaan määrittää isäntä- ja orjalaitteen kellojen keskimääräinen poikkeama kaavan 3 mukaisesti [19].

$$Kellojen\ Poikkema = \frac{(t_2 - t_1) + (t_4 - t_3)}{2} \quad (3)$$

Orjat säätävät kellonsa samaan tahtiin isäntä-kellon kanssa, aivan kuten NTP-protokollassa. Protokollasta on tällä hetkellä tarjolla jo toinen versio, jossa on parannettu protokollan sopivuutta langattomiin laitteisiin, esimerkiksi lähetettävien pakettien koko on pienentynyt [19].

6.5.5 FTSP

FTSP (Flooding Time Syncronization Protocol) on langattomiin sensoriverkkoihin suunnattu aikasynkronointi-protokolla. FTSP-protokolla voidaan saavuttaa erittäin korkeita alle 2 ms tarkkuuksia, mutta tällaisten tarkkuuksien saavuttaminen vaatiikin erityisen MAC-tason aikaleimaustoteutuksen, sekä käytetyn raudan kalibroimisen siten, että indeterministisistä viiveistä päästään eroon. Erittäin tarkkojen aikaleimausten ja viiveiden määritysten lisäksi protokolla korjaa kellojen välisiä vääristymiä käyttäen lineaarista sovitusta. [15] [20]

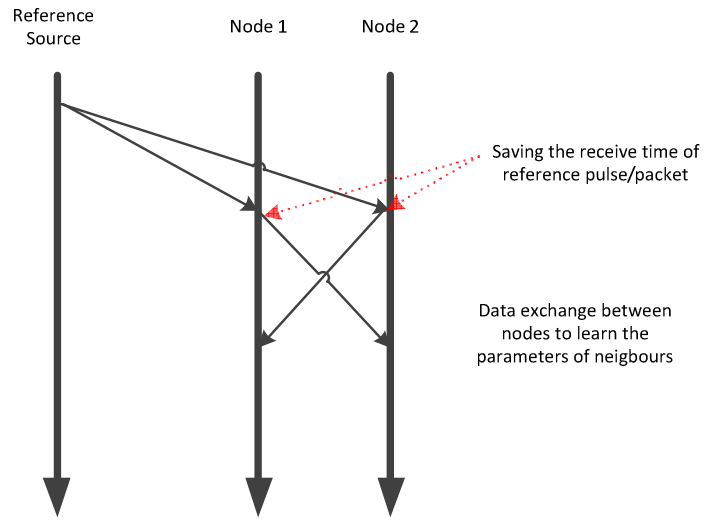
Californian Berkeleyn yliopiston tarjoaa TinyOS-reaaliaikakäyttöjärjestelmän ladattavaksi sivustoltaan BSD-lisenssillä. TinyOS on reaaliaikakäyttöjärjestelmä, jossa FTSP-protokolla on sisäänrakennettuna. TinyOS on sovitettu monille eri prosessoreille ja radio-rajapinnoille, mutta ei ainakaan tässä vaiheessa 802.11-standardin mukaisille laitteille. [20]

6.6 Erilaiset lähestymistavat

Perinteisen lähettäjän ja vastaanottajan välisen synkronoinnin sijaan synkronointi voidaan suorittaa vastaanottajien kesken, kuten esimerkiksi RBS-protokollassa tehdään tai synkronointi voidaan suorittaa näiden tapojen yhdistelmänä, kuten TSync-protokollassa.

6.6.1 RBS

RBS (Reference Broadcast Synchronization Protocol) on monella tavalla erilainen verrattuna tavanomaisiin aikasynkronointiprotokolliin. RBS-protokollassa käytetään synkronoinnin referenssilähteenä referenssi-signaalia, jonka kaikki kanavalla toimivat vastaanottajat voivat vastaanottaa.



Kuva 15. RBS synkronointi mekanismi

RBS-synkronointiprosessi toimii seuraavalla tavalla:

1. RBS-protokollassa yksi verkon asemista lähettää synkronointiviestin kaikille verkon asemille käyttäen fyysistä lähetyskanavaa.
2. Kanavaa kuuntelevat asemat tallentavat referenssipaketin vastaanottohetken oman vapaasti juoksevan laskurinsa kellonaikana.
3. Tämän jälkeen asemat vaihtavat keskenään tietoja vastaanotetuista referenssi paketeista.
4. Tietojen perusteella kellojen aikaleimojen muunnoksille lasketaan parametrit käyttäen liukuvaa keskiarvoa tai lineaarisen regression kaavoja.

RBS-protokollaa käyttämällä päästään eroon lähetysajan ja kanavan hakuajan tuomista virheistä, jolloin virheen aiheuttajiksi jäävät kanavalle ovat signaalin etenemiseen ja vastaanottoon kuluvat ajat.

RBS-protokollaan virhettä aiheuttavat ajat ovat: [21]

- etenemisviive, lyhyillä etäisyyksillä signaalin etenemisviiveen vaikutus on todella pieni, ja otettuna huomioon, että RBS-protokollan tapauksessa vain vastaanottajien etäisyys erolla referenssi pulssin lähettäjään on merkitystä.
- keskeytyksenviive, vastaanoton valmistumisen ja vastaanottokeskeytyksen välinen aika.
- leimauksenviive, keskeytyksestä aikaleiman saamiseen kuluva aika.
- kellon mahdollinen harhailu ennen vertailutietojen vaihtamista.

RBS-protokollan tarkkuus ei kärsi suuresti edes verkon ollessa suuren kuormituksen alaisena, johtuen sen broadcast-tyyppisestä synkronoinnista. RBS-protokollan hyvä puoli on sen postapriorisuus, eli synkronointi voidaan suorittaa aikaleimaamisen jälkeen ja sitten tehdä arvio kellonajasta tuolle menneelle ajankohdalle. [21]

RBS-prokollaan on kehitetty useampia erilaisia parannuksia, alla on lueteltuna muutama parannus jotka olivat kirjoitushetkellä toteutettuja.

- Multihop-RBS on laajennus, jossa verkkojen liitoskohdissa toimii niin sanottu tulkkiasema, joka kuuntelee kahta tai useampaa verkkoa ja suorittavat verkkojen väliset aikakäännökset. [21]
- Efficient RBS on laajennus, jossa lähetettyjen viestien määrää lasketaan muokkaamalla synkronointiviestien lähetyssuhdetta, eli lähetettyjen viestien määrä vähenee ja näin myös verkon asemien virrankulutus laskee. [21]
- Reference Interpolation Protocol eli RIP on RBS-variaatio, jossa verkon referenssipaketin lähettäjän lisäksi toimii erillinen base node, joka toimittaa referenssipaketin vastaanottohetkensä muille nodeille, jolloin verkon nodeilla on kaksi referenssi kelloa joiden avulla ne voivat interpoloida kellonaikansa. Lähetettyjen viestien määrä vähenee huomattavasti ja näin myös verkon toiminta-aikaa on saatu pidemmäksi. [22]

6.6.2 TSync

TSync on aikasynkronointiprotokolla, joka koostuu kahdesta aliprotokollasta, HRTS-protokollasta (Hierarcy Refering Time Synchronization) ja ITR-protokollasta (Individual-based Time Request).

TSync-protokollassa synkronointi tapahtuu seuraavasti:

1. referessinode lähettää muille verkon nodeille sync_begin-paketin, jossa toimitetaan paketin lähetysajankohta.
2. Verkon nodet tallentavat paketin vastaanotto hetken kellonajan.
3. Jokaiseen sync_begin-pakettiin on asetettu yhden verkon noden tunniste, jonka perusteella yksi nodeista vastaa referenssinodelle lähettämällä tiedon sync_begin-paketin vastaanottohetkestä ja answer-paketin lähetys hetkestä.

4. Tämän jälkeen referenssi-node laskee poikkeaman ja lähettää poikkeama tiedon sekä edellisen vastaaja-noden aikaleiman, jonka perusteella nodet muokkaavat kellojaan.

6.7 Aikaleimauksen lähteet Windows-sovelluksille

Windows käyttöjärjestelmässä aikaleimaamiseen on useita erilaisia mahdollisuuksia käytettäessä Microsoft Visual C++ ohjelmointiympäristöä, käytettävissä olevissa aikaleimaus tavoissa on kuitenkin eroja.

6.7.1 Time-funktio

Time-funktion antaman aikaleiman tarkkuus riippuu täysin siitä kuinka funktio on toteutettu. Tämä tarkkuuden epävarmuus taas johtuu siitä, ettei C-kielessä ei ole standardia määritelmää ajanlaskun aloitusajankohdasta, tarkkuudesta, lukualueesta tai koodauksesta.

6.7.2 QueryPerformanceCounter-funktio

QPC-funktion tarkkuus riippuu käytetystä laitteistosta, tavallisesti QPC-laskurin taajuus on tavallisesti prosessorin kellotaajuus tai puolet siitä. QPC-funktion antama lukuarvo on prosessorinkäynnistyksestä kulunut aika laskurin askeleina. Kun tulos halutaan kuluneeksi ajaksi, täytyy QueryPerformanceCounter-funktion tulos jakaa QueryPerfonmanceFrequency-funktion arvolla kaavan 4 mukaisesti. Vaikka funktiolta saadun leimantarkkuus on mikrosekunti-luokkaa, täytyy muistaa, että pelkkä API:n kutsuminen voi kestää 20 mikrosekuntia. [23]

$$\frac{QPC_{value}}{QPC_{freq}} = Kulunut\ aika\ sekuntteina \qquad (4)$$

7 TOTEUTUS JA TULOKSET

Tässä luvussa keskitytään aikaleimauksen ja aikasynkronoinnin toteuttamiseen ja testaamiseen. Aikasynkronointia käsittelevä osa on luottamuksellista tietoa, eikä sen vuoksi ole osa julkista versiota.

7.1 Aikaleimauksen sijoittaminen

Aiemmassa toteutuksessa mittauspakettien leimaustapa oli kaukana 1 ms:n tarkkuudesta, koska pakettien aikaleimaus suoritettiin vasta mittaustulosten lähetyksen yhteydessä radiokortilla, tällä leimaustavalla virhe riippui täysin radiokortin kuormasta.

Mittausten aikaleimauksen toteuttamiseksi oli käytettävissä kaksi erilaista lähestymistapaa:

1. mittauskortin ja radiokortin kellojen pareittaissynkronointi, jolloin leimaaminen toteutettaisiin mittauskortilla
2. mittausajankohdan toimittaminen radiokortille käyttäen keskeytystä, jolloin leimaaminen tapahtuisi radiokortilla.

Mikäli kellot synkronoitaisiin keskenään, voitaisiin käyttää pareittaissynkronointia, jonka tarkkuutta olisi mahdollista parantaa käyttämällä synkronoinnissa esimerkiksi Tiny-Sync-protokollaa. Kellojen pareittaissynkronointi toisi kuitenkin lisää liikennettä korttien välille. Mittausajankohdan toimittaminen radiokortille käyttämällä GPIO-keskeytystä ei lisäisi liikennettä SPI-väylällä, eikä mittauskortilla tarvitsisi suorittaa suoritusaikaa vieviä toimintoja, vaan yhden linjan käyttäminen alhaalla mittauksen yhteydessä riittäisi.

7.2 Aikaleimauksen toteuttaminen

Keskeytysten käyttöönotto ei ollut aluksi helppoa, koska kokemukseni reaaliaikakäyttöjärjestelmistä oli vähäistä. Jouduin turvautumaan G2 Microsystemsin tekniseen tukeen, koska GPIO-keskeytyksistä ei ollut tarjolla ohjeita ja ainoana tapana selvittää keskeytysten käyttöä oli lukea lähdekoodia.

Aluksi sain keskeytykset toimimaan muokkaamalla suoraan rekisteriä, mutta muokkausrivit jäivät pois turhina uudemman G2C501 SDK:n version myötä. Uudessa SDK:ssa GPIO-keskeytykset olivat selkeämmin opastetut ja keskeytysten käyttöön ottoon oli tarjolla uudet omat funktionsa.

Kuten kuvasta 16 on nähtävillä, G2 Microsystemsin GPIO-keskeytystyskäsittely on kolmitasoinen. Matalin taso on ISR-taso (Interrupt Service Routine), jossa tavallisesti suoritetaan ainoastaan korkeamman tason DSR-keskeytyksen (Deferred Service Routine) luominen havaitusta keskeytysvektorista. DSR taas luo tapahtuman (Event) G2 Microsystemsin käyttämään tapahtumapinoon. Tapahtuma suoritetaan, kun sen vuoro tapahtumapinoa luettaessa saavutetaan.

Kuva 16. Keskeytyskäsittely

GPIO-keskeyksestä oli siis mahdollista luoda tapahtuma, joka suoritettaisiin aina keskeytyksen tapahtuessa. Aikaleimaus täytyi kuitenkin suorittaa matalammalle tasolle, koska tapahtumatasolla paketin aikaleima päästäisiin kirjoittamaan vasta tapahtuman tullessa suoritusvuoroon.

Olin tutustunut G2 SDK:n mukana toimitettuun patch-kansioon, joka sisältää valmiita pohjia erikoisempien muutosten lisäämiseksi ohjelmiin. G2C501:n sovellusohjelmoijanoppaassa mainitaan kuitenkin ainoastaan, että käytettävät paikkaukset otetaan käyttöön lisäämällä ennalta määrättyjä vakioita sovelluksen käynnistyskoodin sisältävään appstart.c-tiedostoon. Käynnistystiedostoon lisättävät koodirivit korjasivat ainoastaan G2C501-piirin ROM-muistissa olevan eCos-käyttöjärjestelmän sisältämiä virheitä ja ominaisuuksia.

Tarkempien tiedustelujen jälkeen sain G2 Microsystemisiltä vastauksen koskien ISR-käsittelykoodin muokkaamista. Koodin käyttöönottamiseksi tarvitsi ainoastaan kopioida patch-kansiossa sijaitseva gpio_intr.c-tiedosto sovelluksen kansioon ja lisätä kääntäjän antama gpio_intr.o-objektitiedosto linkitettäväksi mukaan sovellukseen. Muokattujen GPIO-keskeytysten lisääminen ei siis vaatinut minkäänlaisia muutoksia itse pääsovellukseen; ainoastaan sovelluksen Makefile-tiedosto tarvitsi tiedon linkitettävästä tiedostosta.

Muokattu ISR-käsittely mahdollisti huomattavasti tarkemman aikaleimaamisen, koska tapahtumapinon kuormalla ei ollut enää vaikutusta leimaamisen tarkkuuteen.

8 YHTEENVETO

Aikasynkronoinnin tarkkuus yhden tukiaseman toteutuksella on nyt tulosten perusteella alle vaaditun 1 ms:n. Mittaustuloksia on siis mahdollista käyttää ajan suhteen synkronisten analyysien lähteenä.

Käytetty synkronointitapa tarvitsee kuitenkin lisävarmennusta, jotta synkronointiprosessin tarkkuuteen voitaisiin luottaa. Aikaleimauksen tarkkuus on kriittisessä osassa langattomissa järjestelmissä ja korkean tarkkuuden saavuttaminen vaatii yleensä hardware-tason aikaleimaustoteutuksen.

Tulevaisuudessa toteutuksen tarkkuutta voitaisiin parantaa edelleen muokkaamalla vastaanoton aikaleimausta lähemmäksi fyysistä rajapintaa. Reaaliaikaisen tiedonsiirron käyttöönottamiseksi SPI-tiedonsiirto olisi hyvä sijoittaa omaan säikeeseensä.

LÄHTEET

1. Honeywell International. *Machine Sentinel Solution Note*. Honeywell. [online] [Viitattu 28. 3 2011.] Saatavissa: <http://hpsweb.honeywell.com/NR/rdonlyres/CD5AA356-5212-4224-A1D1-14254614A977/55559/PBR30400101.pdf>

2. Monarc instruments. Synchronous Time Averaging. [online] [Viitattu: 28. 03 2011.] Saatavissa: <http://www.monarchinstrument.com/pdfs/synctimeavg.pdf>

3. G2 Microsystems. *G2C501 SoC. Product Brief*. [online] 2009. [Viitattu: 05. 04 2011.] Saatavissa: <http://www.g2microsystems.com/downloads/PB501.pdf>

4. eCosCentric Limited. eCos. Homepage. [online] 2011. [Viitattu: 28. 3 2011.] Saatavissa: <http://ecos.sourceware.org/>

5. Texas Instruments Incorporated. MSP430F15x, MSP430F16x, MSP430F161x Mixed signal Microcontroller datasheet. [online] 2002. [Viitattu: 6. 4 2011.]Saatavissa: <http://focus.ti.com/lit/ds/slas368g/slas368g.pdf. SLAS368G>

6. Labrosse, Jean J. *MicroC/OS-II The Realtime Kernel (Second Edition)*. CMP Books, 1998. ISBN 0879305436.

7. Micriμm Technologies Corporation. Micrium. Homepage. [online] 2011. [Viitattu: 28. 03 2010.] Saatavissa: <http://micrium.com>

8. Honeywell International Inc. *Honeywell Wireless Solutions esite*. BR-06-08-ENG. Phoenix : Honeywell International Inc., 2007. BR-06-08-ENG.

9. IEEE Standards Association. *IEEE802.11a. High-speed Physical Layer in the 5 GHz Band.* [online] [Viitattu: 6. 4. 2011] Saatavissa: <http://standards.ieee.org/getieee802/download/802.11a-1999.pdf>

10. IEEE Standards Association. *802.11b. Higher-Speed Physical Layer Extension in the 2.4 GHz Band.* [online]
[Viitattu: 6. 4. 2011]
Saatavissa: <http://standards.ieee.org/getieee802/download/802.11b-1999.pdf>

11. IEEE Standards Association. *IEEE802.11g. Further Higher Data Rate Extension.* [online]
[Viitattu: 6. 4. 2011]
Saatavissa: <http://standards.ieee.org/getieee802/download/802.11g-2003.pdf>

12. IEEE Standards Association. *IEEE802.11n-2009.* [online] 2009.
[Viitattu: 28. 03. 2011] Saatavissa:
<http://standards.ieee.org/getieee802/download/802.11n-2009.pdf>

13. IEEE Standards Association. *IEEE802.11-2007.* [Online] 2007.
[Viitattu: 28. 03.2011] Saatavissa:
<http://standards.ieee.org/getieee802/download/802.11-2007.pdf>

14. Kaario, Kimmo. *TCP/IP-verkot*. Docendo, 2002. s. 156. ISBN 9518461074.

15. Holger, Karl ja Willig, Andreas. *Protocols and Architectures for Wireless Sensor Networks*. Wiley, 2005. ISBN 0470095105.

16. Elson, Jeremy. *Time Synchronization in Wireless Sensor Networks*.
Jeremy Elson's Writings. [online] 2003.
[Viitattu: 4. 4. 2011]
Saatavissa: <http://lecs.cs.ucla.edu/~jelson/dissertation-final.pdf>

17. NTP Project. The Network Time Protocol Distribution. [online]
[Viitattu: 6. 4. 2011] Saatavissa:
<http://www.eecis.udel.edu/~mills/ntp/html/index.html.>

18. Network Time Synchronization Research Project. Future Plans. [Online]
[Viitattu: 28. 03. 2011] Saatavissa:
<http://www.eecis.udel.edu/~mills/ntp.html>

19. Agilent Technologies, Inc. *IEEE1588 Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems - Tutorial*. The National Institute of Standards and Technology. [online] [Viitattu: 4. 4. 2011] Saatavissa: <http://www.nist.gov/el/isd/ieee/upload/tutorial-basic.pdf>

20. Maroti, Miklos ja Kusy, Brano. *TinyOS Documentation Wiki*. FTSP. [Online] 2008. [Viitattu: 6. 4 2011.] Saatavissa: <http://docs.tinyos.net/index.php/FTSP>

21. Hyojung Lee, Wonpil Yu, Youngmi Kwon. *Efficient RBS in Sensor Networks*. Institute of Electrical and Electronics Engineers, 2006.

22. Youngtae Jo, Chongmyung Park, Joahyoung Lee, Inbum Jung. Energy *Effective Time Synchronization in Wireless Sensor Network*. Institute of Electrical and Electronics Engineers, 2007.

23. Microsoft. Support. *How To Use QueryPerformanceCounter to Time Code*. [online] 2011. [Viitattu: 28. 03.2011] Saatavissa: <http://support.microsoft.com/kb/172338>

www.savonia.fi

Savonia
ammattikorkeakoulu